This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims

1. (Currently amended) A method for implementing a remote method call, comprising:

generating remote objects;

generating at least one proxy object, where each proxy object corresponds to one remote object;

including data from the remote object into the proxy object; and

processing a call to a method on one proxy object~~; and~~ by:

(i) determining whether the method is implemented locally;

(ii) executing a local method including code to perform method operations on the proxy object without going to the remote object, in response to determining that the method is implemented locally; and

(iii) executing a remote method including code to perform method operations on the remote object, in response to determining that the method is not implemented locally.

~~executing the method, wherein the method is one of a plurality of methods, wherein at least one of the plurality of methods comprises a local method including code to perform method operations on the proxy object without going to the remote object and at least one other of the plurality of methods comprises a remote method including code to perform method operations on the remote object.~~

2. (Currently amended) The method of claim 1, wherein executing one local method comprises processing data in the proxy object from the remote object, wherein remote object data is returned to ~~the~~ a caller from the proxy object, wherein the remote object data is the data from the remote object included into the proxy object.

3. (Original) The method of claim 1, wherein generating the at least one proxy object further comprises:

transmitting a request to instantiate a proxy object for one specified remote object and a specified proxy class;

generating the proxy object in response to the request;

calling an initialization method from the specified proxy class to add data from the specified remote object to the generated proxy object; and

returning the generated proxy object for use in accessing the remote object.

4. (Original) The method of claim 3, wherein the proxy object class is a subclass of a remote object class, wherein the remote objects are instantiated from at least one remote object class, and wherein each remote object class is a subclass of a communication protocol class providing methods and attributes.

5. (Original) The method of claim 1, further comprising:

generating a client communication object; and

generating a server communication object, wherein the client and server communication objects enable communication therebetween.

6. (Original) The method of claim 5, wherein one client communication object is generated for each server communication object to which the client communication object will communicate.

7. (Original) The method of claim 5, wherein each client communication object is instantiated from a client communication class and wherein each server communication object is instantiated from a server communication class, wherein the client and server communication classes implement methods from a communication protocol class.

8. (Currently amended) The method of claim 7, wherein the communication protocol class comprises ~~the~~ a Java Remote Method Invocation (RMI) classes.

9. (Original) The method of claim 7, wherein the client communication object manages communication with the server communication object for a plurality of proxy objects on which remote methods are invoked to execute on corresponding remote objects.

10. (Original) The method of claim 7, wherein the client communication class includes code to handle error exceptions generated from the communication protocol class in response to executing remote methods on the plurality of proxy objects.

11. (Currently amended) The method of claim 5, wherein the server communication object comprise a first server communication object~~, respectively~~, and wherein executing one remote method further comprises:

passing the remote method to the client communication object;

transmitting to the server communication object, with the client communication object, an invocation method specifying the remote method on one specified remote object to the server communication object;

determining, with the server communication object, whether the remote object specified in the received invocation method is accessible through a second server communication object; and

transmitting, with the first server communication object, the received invocation method to the second server communication object to execute against the specified remote object.

12. (Original) The method of claim 11, further comprising:

executing, with the second server communication object, the remote method specified in the invocation method on the specified remote object;

returning, with the second server communication object, data generated in response to execution of the remote method on the specified remote object to the client communication object.

13. (Original) The method of claim 5, wherein the client communication class includes code to handle exceptions related to communication between the client communication object and the server communication object.

14. (Original) The method of claim 1, wherein the remote object and proxy object are implemented in a same computer device, and wherein the local and remote method perform their operations on the same computer device.

15. (Original) The method of claim 1, wherein the remote objects are generated on a server system, wherein the call to the method on one proxy object occurs on a client system, wherein the server and client systems communicate over a network, wherein the local method includes code to perform the method operation on the proxy object on the client without going to the server and the remote method includes code to execute remotely on the server to perform method operations on the remote object on the server.

16. (Original) The method of claim 15, wherein the server comprises a first server and wherein the called method comprises a remote method, further comprising:

transmitting, with the client, information indicating the called remote method and remote object corresponding to the proxy object subject on which the remote method is called;

receiving, with the first server, the information indicating the remote method and remote object to execute;

determining, at the first server, whether the indicated remote object is on a second server; and transmitting information on the indicated remote method and remote object to a second server for execution thereon if the indicated remote object for execution on the second server.

17. (Original) The method of claim 16, further comprising:

launching, with the client, an applet downloaded from the first server, wherein the applet calls the remote method on the proxy object corresponding to the remote object on the second server, and wherein the applet communicates the information indicating the remote method and remote object to the first server.

18. (Original) The method of claim 14, wherein the server comprises a first server and wherein generating the at least one proxy object further comprises:

receiving, with the first server, a method to instantiate a proxy object for a specified remote object;

determining, at the first server, whether the specified remote object is on a second server;

transmitting, with the first server, the method to instantiate the proxy object to the second server for execution thereon if the remote object is located on the second server.

19. (Original) The method of claim 18, further comprising:

returning, with the second server, the instantiated proxy object to the first server to return to the client.

20. (Currently amended) A system for implementing a remote method call, comprising:

means for generating remote objects;

means for generating at least one proxy object, where each proxy object corresponds to one remote object;

means for including data from the remote object into the proxy object; <u>and</u>

means for processing a call to a method on one proxy object~~; and~~ by:

(i) determining whether the method is implemented locally;

(ii) executing a local method including code to perform method operations on the proxy object without going to the remote object, in response to determining that the method is implemented locally; and

(iii) executing a remote method including code to perform method operations on the remote object, in response to determining that the method is not implemented locally.

~~means for executing the method, wherein the method is one of a plurality of methods, wherein at least one of the plurality of methods comprises a local method including code to perform method operations on the proxy object without going to the remote object and at least one other of the plurality of methods comprises a remote method including code to perform method operations on the remote object.~~

21. (Currently amended) The system of claim 20, wherein the means for executing one local method comprises processing data in the proxy object from the remote object, wherein remote object data is returned to ~~the~~ a caller from the proxy object, wherein the remote object data is the data from the remote object included into the proxy object.

22. (Original) The system of claim 20, wherein the means for generating the at least one proxy object further performs:

transmitting a request to instantiate a proxy object for one specified remote object and a specified proxy class;

generating the proxy object in response to the request;

calling an initialization method from the specified proxy class to add data from the specified remote object to the generated proxy object; and

returning the generated proxy object for use in accessing the remote object.

23. (Original) The system of claim 22, wherein the proxy object class is a subclass of the a remote object class, wherein the remote objects are instantiated from at least one remote object class, and wherein each remote object class is a subclass of a communication protocol class providing methods and attributes.

24. (Original) The system of claim 20, further comprising:

means for generating a client communication object; and

means for generating a server communication object, wherein the client and server communication objects enable communication therebetween.

25. (Original) The system of claim 24 wherein one client communication object is generated for each server communication object to which the client communication object will communicate.

26. (Original) The system of claim 24, wherein each client communication object is instantiated from a client communication class and wherein each server communication object is instantiated from a server communication class, wherein the client and server communication classes implement methods from a communication protocol class.

27. (Currently amended) The system of claim 26, wherein the communication protocol class comprises ~~the~~ a Java Remote Method Invocation (RMI) classes.

28. (Original) The system of claim 26, wherein the client communication object manages communication with the server communication object for a plurality of proxy objects on which remote methods are invoked to execute on corresponding remote objects.

29. (Original) The system of claim 26, wherein the client communication class includes code to handle error exceptions generated from the communication protocol class in response to executing remote methods on the plurality of proxy objects.

30. (Currently amended) The system of claim 24, wherein the server communication object comprise a first server communication object~~, respectively~~, and wherein the means for executing one remote method further performs:

passing the remote method to the client communication object;

transmitting to the server communication object, with the client communication object, an invocation method specifying the remote method on one specified remote object;

determining, with the server communication object, whether the remote object specified in the received invocation method is accessible through a second server communication object; and

transmitting, with the first server communication object, the received invocation method to the second server communication object to execute against the specified remote object.

31. (Original) The system of claim 30, further comprising:

means for executing, with the second server communication object, the remote method specified in the invocation method on the specified remote object;

means for returning, with the second server communication object, data generated in response to execution of the remote method on the specified remote object to the client communication object.

32. (Original) The system of claim 24, wherein the client communication class includes code to handle exceptions related to communication between the client communication object and the server communication object.

33. (Original) The system of claim 20, further comprising:

a computer device, wherein the remote object and proxy object are implemented on the computer device.

34. (Original) The system of claim 20, further comprising:

a client system;

a server system;

a network for enabling communication between the client and server system, wherein the remote objects are generated on the server system, wherein the call to the method on one proxy object occurs on a client system, wherein the local methods include code to perform the method operation on the proxy object on the client system without going to the server system and the remote methods include code to execute remotely on the server to perform method operations on the remote object on the server.

35. (Currently amended) An article of manufacture including code for implementing a remote method call, wherein the code causes operations to be performed comprising:

generating remote objects;

generating at least one proxy object, where each proxy object corresponds to one remote object;

including data from the remote object into the proxy object; and

processing a call to a method on one proxy object~~; and~~ by:

(i) determining whether the method is implemented locally;

(ii) executing a local method including code to perform method operations on the proxy object without going to the remote object, in response to determining that the method is implemented locally; and

(iii) executing a remote method including code to perform method operations on the remote object, in response to determining that the method is not implemented locally.

~~executing the method, wherein the method is one of a plurality of methods, wherein at least one of the plurality of methods comprises a local method including code to perform method operations on the proxy object without going to the remote object and at least one other of the plurality of methods comprises a remote method including code to perform method operations on the remote object.~~

36. (Currently amended) The article of manufacture of claim 35, wherein executing one local method comprises processing data in the proxy object from the remote object, wherein remote object data is returned to ~~the~~ <u>a</u> caller from the proxy object<u>, wherein the remote object data is the data from the remote object included into the proxy object</u>.

37. (Original) The article of manufacture of claim 35, wherein generating the at least one proxy object further comprises:

transmitting a request to instantiate a proxy object for one specified remote object and a specified proxy class;

generating the proxy object in response to the request;

calling an initialization method from the specified proxy class to add data from the specified remote object to the generated proxy object; and

returning the generated proxy object for use in accessing the remote object.

38. (Original) The article of manufacture of claim 37, wherein the proxy object class is a subclass of a remote object class, wherein the remote objects are instantiated from at least one remote object class, and wherein each remote object class is a subclass of a communication protocol class providing methods and attributes.

39. (Original) The article of manufacture of claim 35, further comprising:

generating a client communication object; and generating a server communication object, wherein the client and server communication objects enable communication therebetween.

40. (Original) The article of manufacture of claim 39, wherein one client communication object is generated for each server communication object to which the client communication object will communicate.

41. (Original) The article of manufacture of claim 39, wherein each client communication object is instantiated from a client communication class and wherein each server communication object is instantiated from a server communication class, wherein the client and server communication classes implement methods from a communication protocol class.

42. (Currently amended) The article of manufacture of claim 41, wherein the communication protocol class comprises ~~the~~ a Java Remote Method Invocation (RMI) classes.

43. (Original) The article of manufacture of claim 41, wherein the client communication object manages communication with the server communication object for a plurality of proxy objects on which remote methods are invoked to execute on corresponding remote objects.

44. (Original) The article of manufacture of claim 41, wherein the client communication class includes code to handle error exceptions generated from the communication protocol class in response to executing remote methods on the plurality of proxy objects.

45. (Currently amended) The article of manufacture of claim 39, wherein the server communication object comprise a first server communication object~~, respectively~~, and wherein executing one remote method further comprises:

passing the remote method to the client communication object;

transmitting to the server communication object, with the client communication object, an invocation method specifying the remote method on one specified remote object;

determining, with the server communication object, whether the remote object specified in the received invocation method is accessible through a second server communication object; and

transmitting, with the first server communication object, the received invocation method to the second server communication object to execute against the specified remote object.

46. (Original) The article of manufacture of claim 45, further comprising:

executing, with the second server communication object, the remote method specified in the invocation method on the specified remote object;

returning, with the second server communication object, data generated in response to execution of the remote method on the specified remote object to the client communication object.

47. (Original) The article of manufacture of claim 39, wherein the client communication class includes code to handle exceptions related to communication between the client communication object and the server communication object.

48. (Currently amended) A method for accessing data from remote objects, comprising:

receiving at least one proxy object, where each proxy object corresponds to one remote object, wherein the proxy object includes data from the remote object; and

processing a call to a method on one proxy object~~; and~~ by:

(i) determining whether the method is implemented locally;

(ii) executing a local method including code to perform method operations on the proxy object without going to the remote object, in response to determining that the method is implemented locally; and

(iii) executing a remote method including code to perform method operations on the remote object, in response to determining that the method is not implemented locally.

~~executing the method, wherein the method is one of a plurality of methods, wherein at least one of the plurality of methods comprises a local method including code to perform method operations on the proxy object without going to the remote object and at least one other of the plurality of methods comprises a remote method including code to perform method operations on the remote object.~~

49. (Currently amended) The method of claim 48, wherein executing one local method comprises processing data in the proxy object from the remote object, wherein remote object data is returned to ~~the~~ a caller from the proxy object, wherein the remote object data is the data from the remote object included into the proxy object.

50. (Original) The method of claim 48, further comprising:

generating a client communication object capable of communicating with a server communication object.

51. (Original) The method of claim 50, wherein one client communication object is generated for each server communication object to which the client communication object will communicate.

52. (Original) The method of claim 50, wherein each client communication object is instantiated from a client communication class and wherein each server communication object is

instantiated from a server communication class, wherein the client and server communication classes implement methods from a communication protocol class.

53. (Currently amended) The method of claim 50, wherein the server communication object comprises a first server communication object, ~~respectively,~~ and wherein executing one remote method further comprises:

passing the remote method to the client communication object;

transmitting to the server communication object, with the client communication object, an invocation method specifying the remote method on one specified remote object to the server communication object, wherein the remote method is capable of being accessible through a second server communication object.

54. (Currently amended) A system for accessing data from remote objects, comprising:

means for receiving at least one proxy object, where each proxy object corresponds to one remote object, wherein the proxy object includes data from the remote object;

means for processing a call to a method on one proxy object; and

means for executing the method by:

(i) determining whether the method is implemented locally;

(ii) executing a local method including code to perform method operations on the proxy object without going to the remote object, in response to determining that the method is implemented locally; and

(iii) executing a remote method including code to perform method operations on the remote object, in response to determining that the method is not implemented locally. ~~, wherein the method is one of a plurality of methods, wherein at least one of the plurality of methods comprises a local method including code to perform method operations on the proxy~~

~~object without going to the remote object and at least one other of the plurality of methods comprises a remote method including code to perform method operations on the remote object.~~

55. (Currently amended) The system of claim 54, wherein the means for executing one local method comprises processing data in the proxy object from the remote object, wherein remote object data is returned to ~~the~~ a caller from the proxy object, wherein the remote object data is the data from the remote object included into the proxy object.

56. (Original) The system of claim 54, further comprising:

means for generating a client communication object capable of communicating with a server communication object.

57. (Original) The system of claim 56, wherein one client communication object is generated for each server communication object to which the client communication object will communicate.

58. (Original) The system of claim 56, wherein each client communication object is instantiated from a client communication class and wherein each server communication object is instantiated from a server communication class, wherein the client and server communication classes implement methods from a communication protocol class.

59. (Currently amended) The system of claim 56, wherein the server communication object comprises a first server communication object, ~~respectively,~~ and wherein the means for executing one remote method further comprises:

passing the remote method to the client communication object;

transmitting to the server communication object, with the client communication object, an invocation method specifying the remote method on one specified remote object to the server communication object, wherein the remote method is capable of being accessible through a second server communication object.

60. (Currently amended) An article of manufacture including code for accessing data from remote objects, wherein the code is capable of causing device operations comprising:

receiving at least one proxy object, where each proxy object corresponds to one remote object, wherein the proxy object includes data from the remote object;

processing a call to a method on one proxy object; and

executing the method by:

(i) determining whether the method is implemented locally;

(ii) executing a local method including code to perform method operations on the proxy object without going to the remote object, in response to determining that the method is implemented locally; and

(iii) executing a remote method including code to perform method operations on the remote object, in response to determining that the method is not implemented locally. ~~, wherein the method is one of a plurality of methods, wherein at least one of the plurality of methods comprises a local method including code to perform method operations on the proxy object without going to the remote object and at least one other of the plurality of methods comprises a remote method including code to perform method operations on the remote object.~~

61. (Currently amended) The article of manufacture of claim 60, wherein executing one local method comprises processing data in the proxy object from the remote object, wherein remote object data is returned to ~~the~~ a caller from the proxy object, wherein the remote object data is the data from the remote object included into the proxy object.

62. (Original) The article of manufacture of claim 60, further comprising:

generating a client communication object capable of communicating with a server communication object.

63. (Original) The article of manufacture of claim 62, wherein one client communication object is generated for each server communication object to which the client communication object will communicate.

64. (Original) The article of manufacture of claim 62, wherein each client communication object is instantiated from a client communication class and wherein each server communication object is instantiated from a server communication class, wherein the client and server communication classes implement methods from a communication protocol class.

65. (Currently amended) The article of manufacture of claim 62, wherein the server communication object comprises a first server communication object, ~~respectively,~~ and wherein executing one remote method further comprises:

passing the remote method to the client communication object;

transmitting to the server communication object, with the client communication object, an invocation method specifying the remote method on one specified remote object to the server communication object, wherein the remote method is capable of being accessible through a second server communication object.